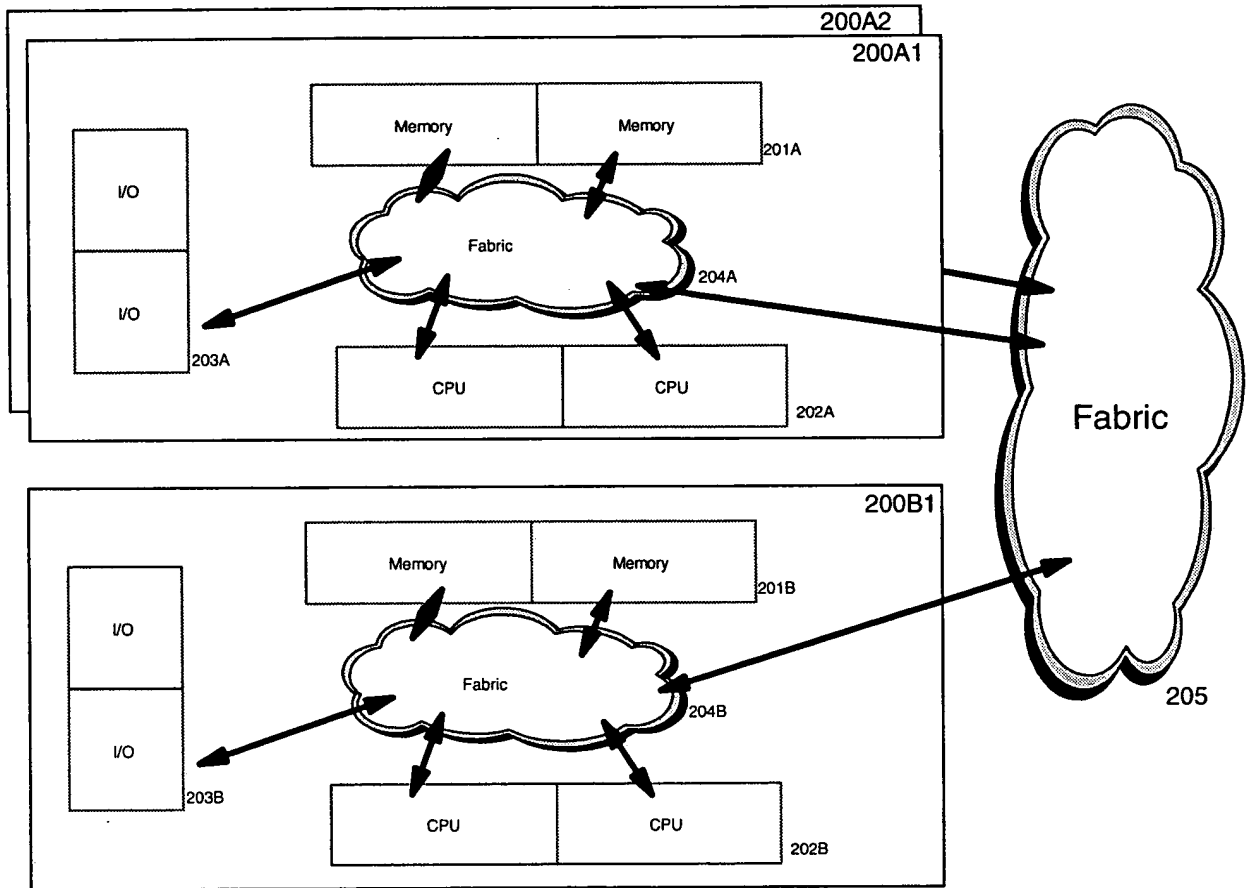
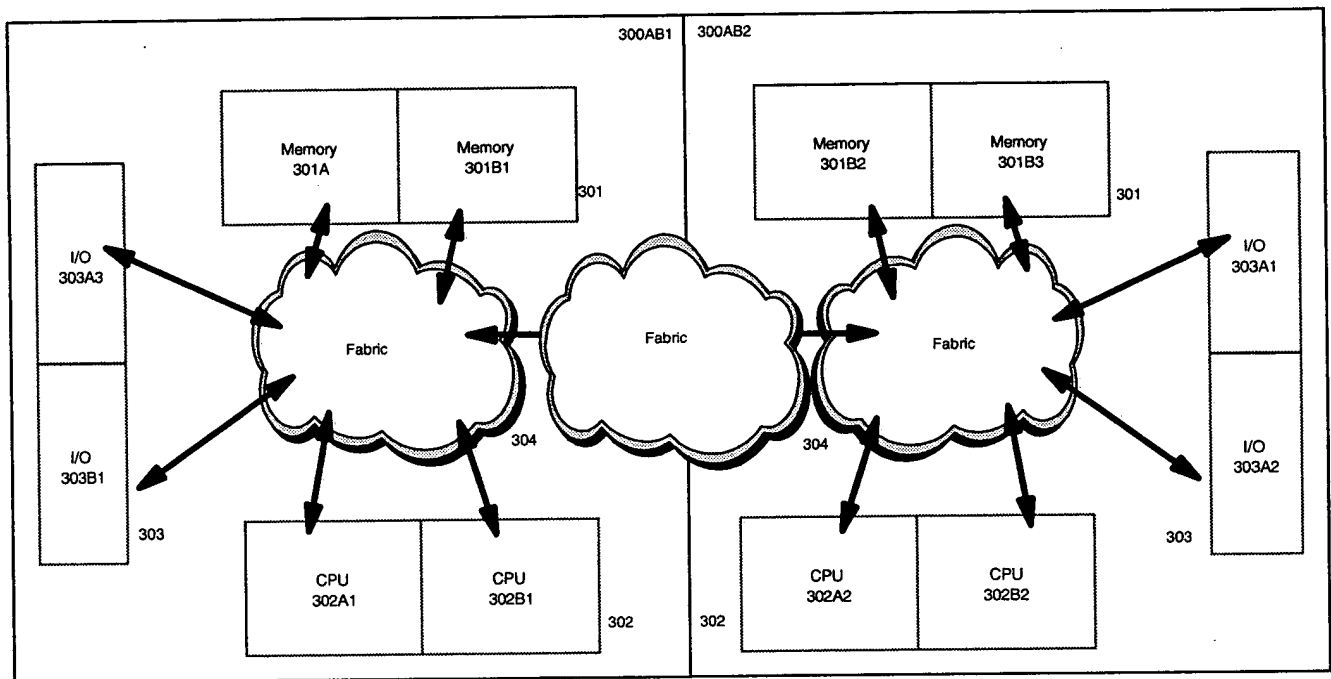
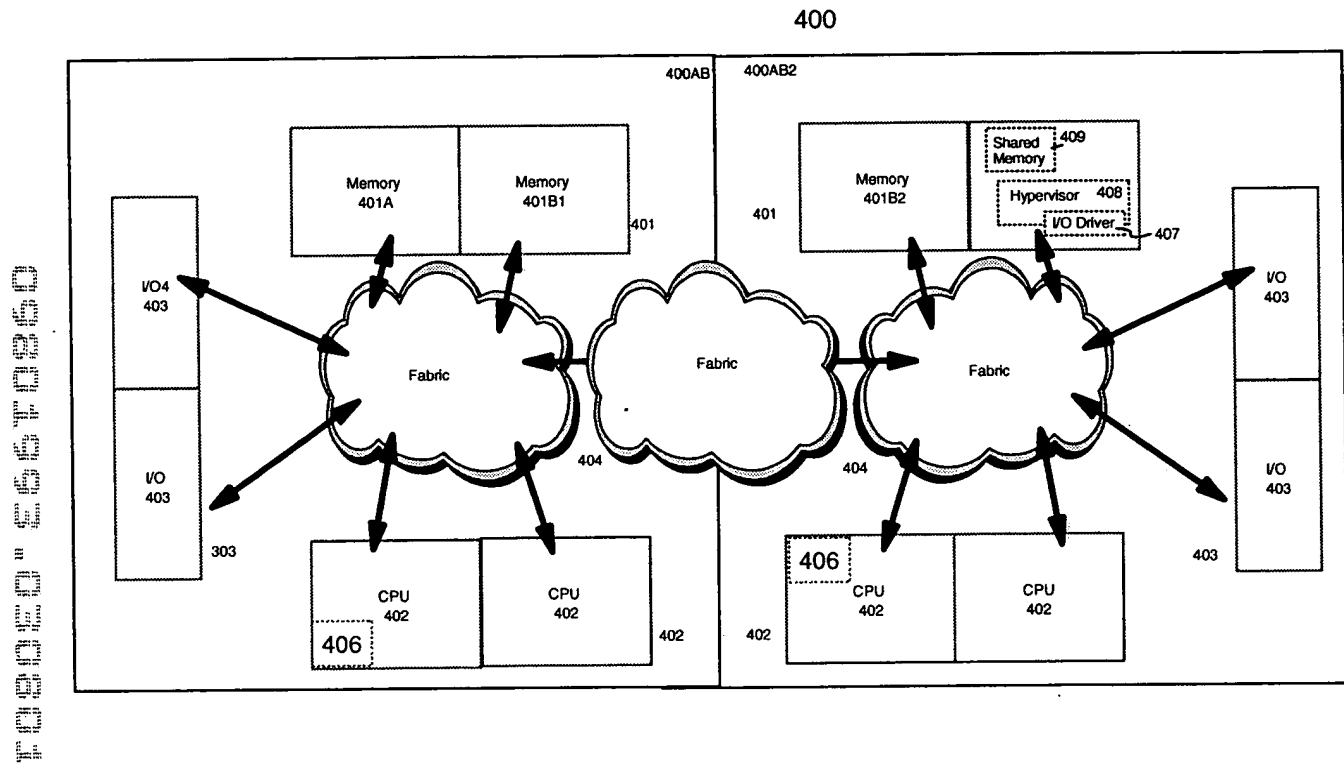


Fig 1

**Fig 2**

**Fig 3**

Virtualization allows sharing of CPUs and I/O elements by multiple partitions



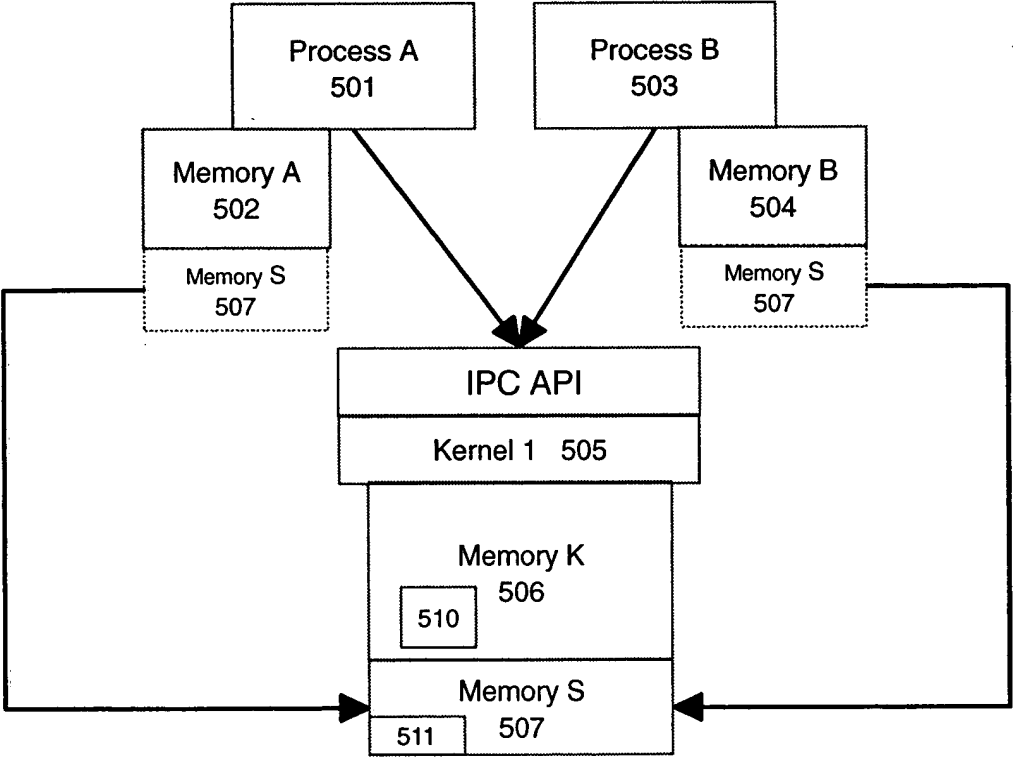
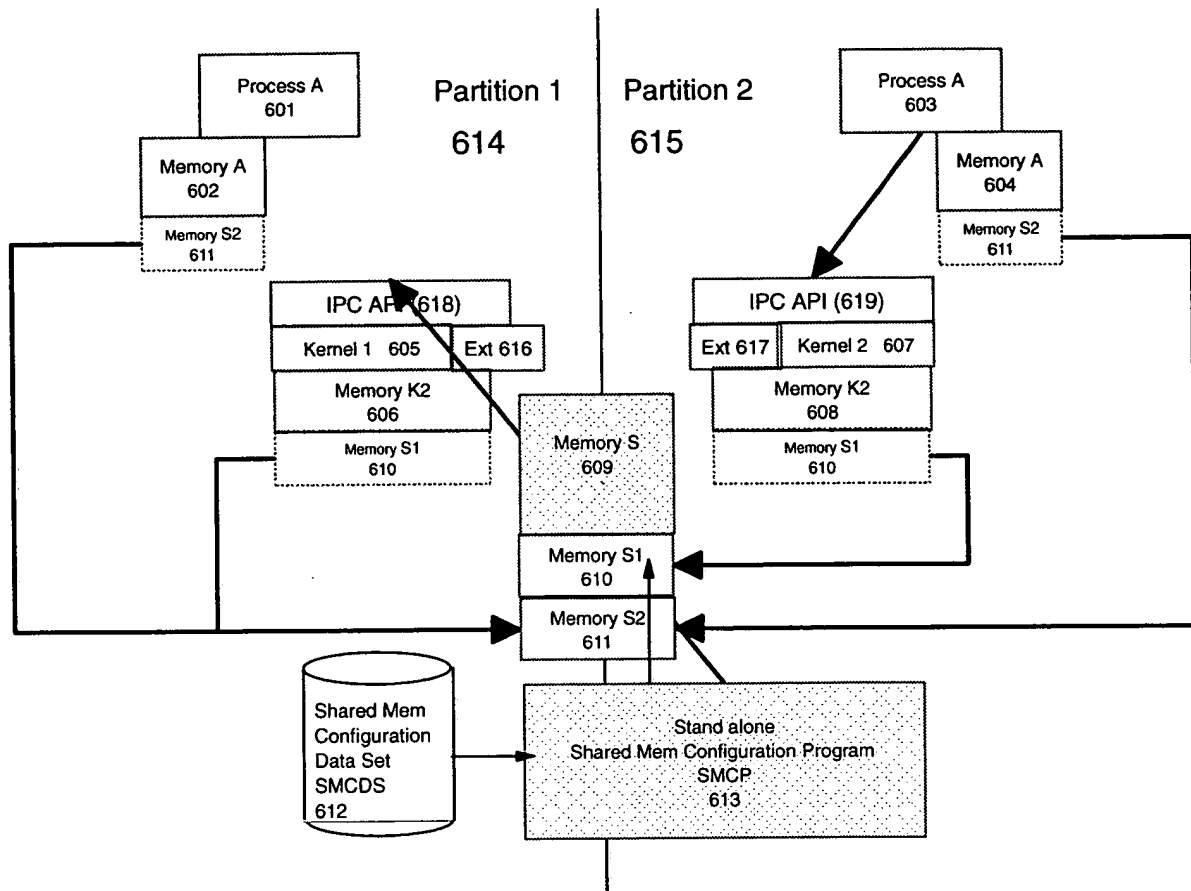
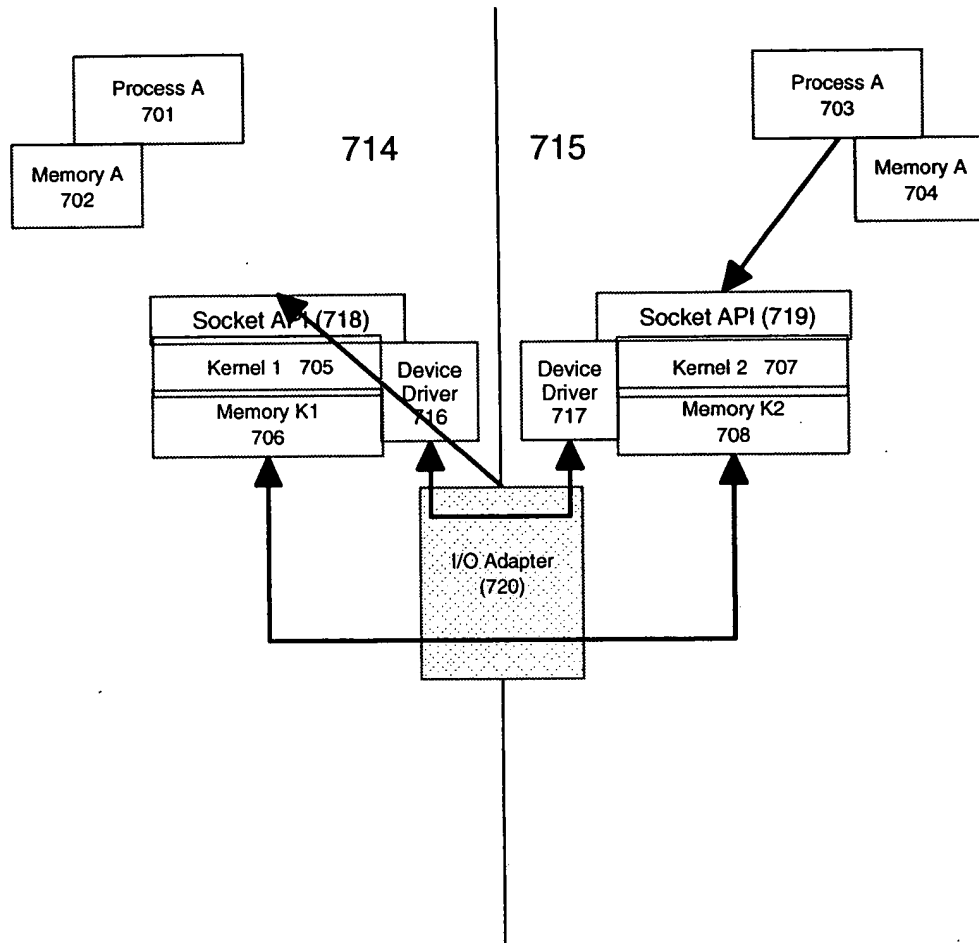
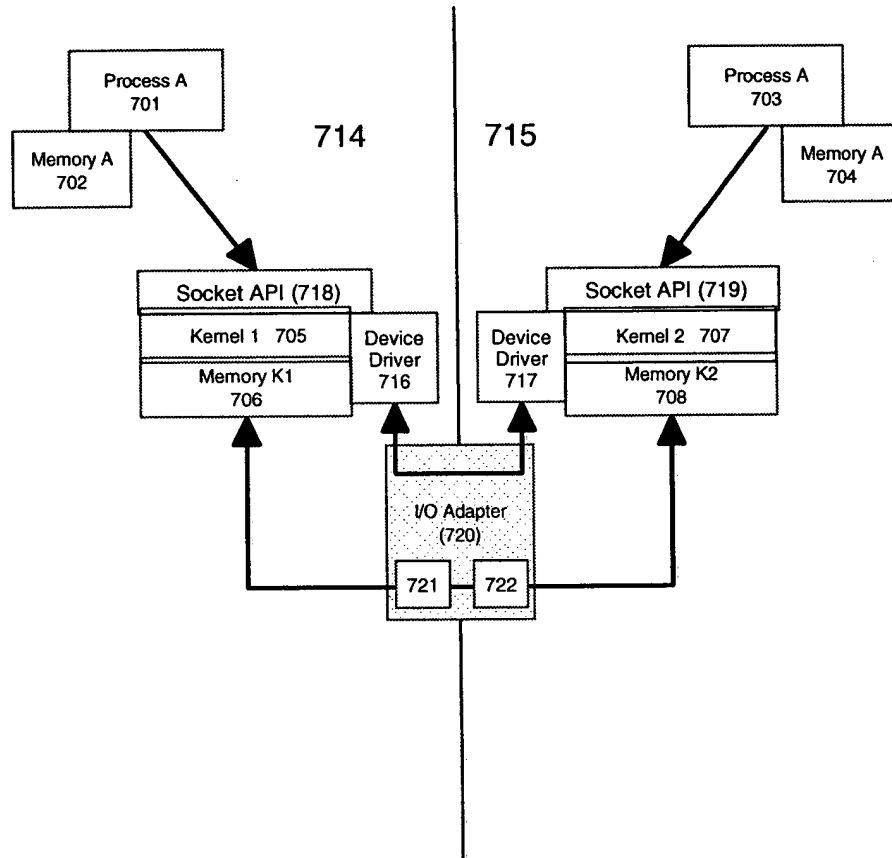


Fig 5

**Fig 6**

**Fig 7A**



The Prior Art
Fig 7B

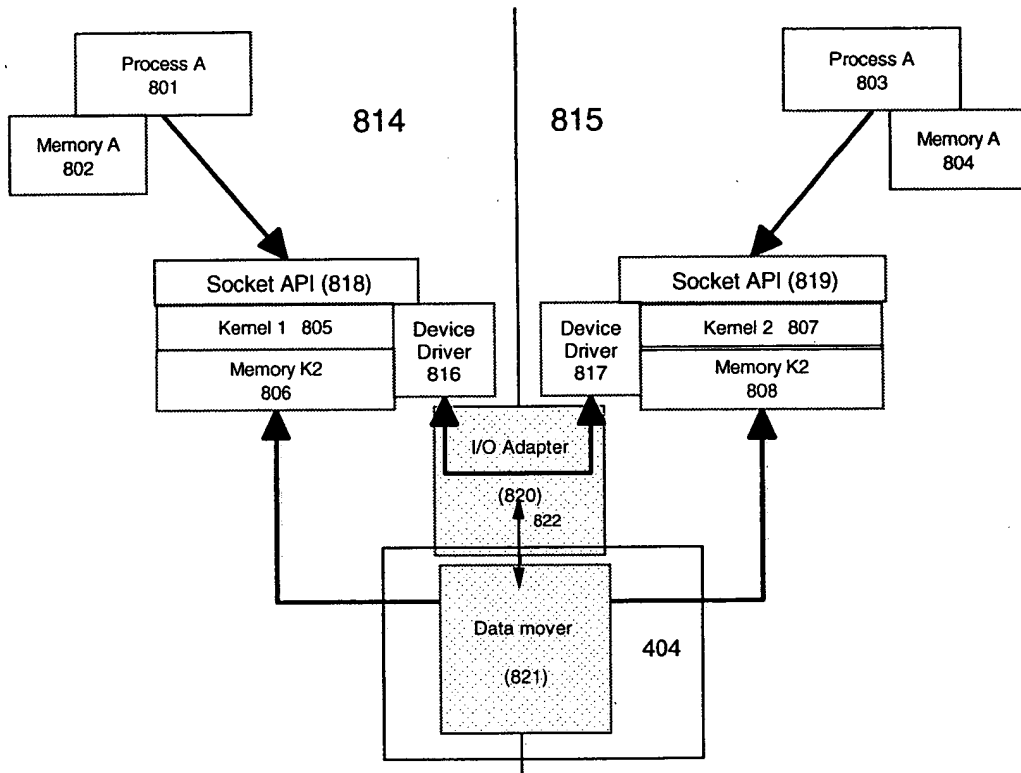
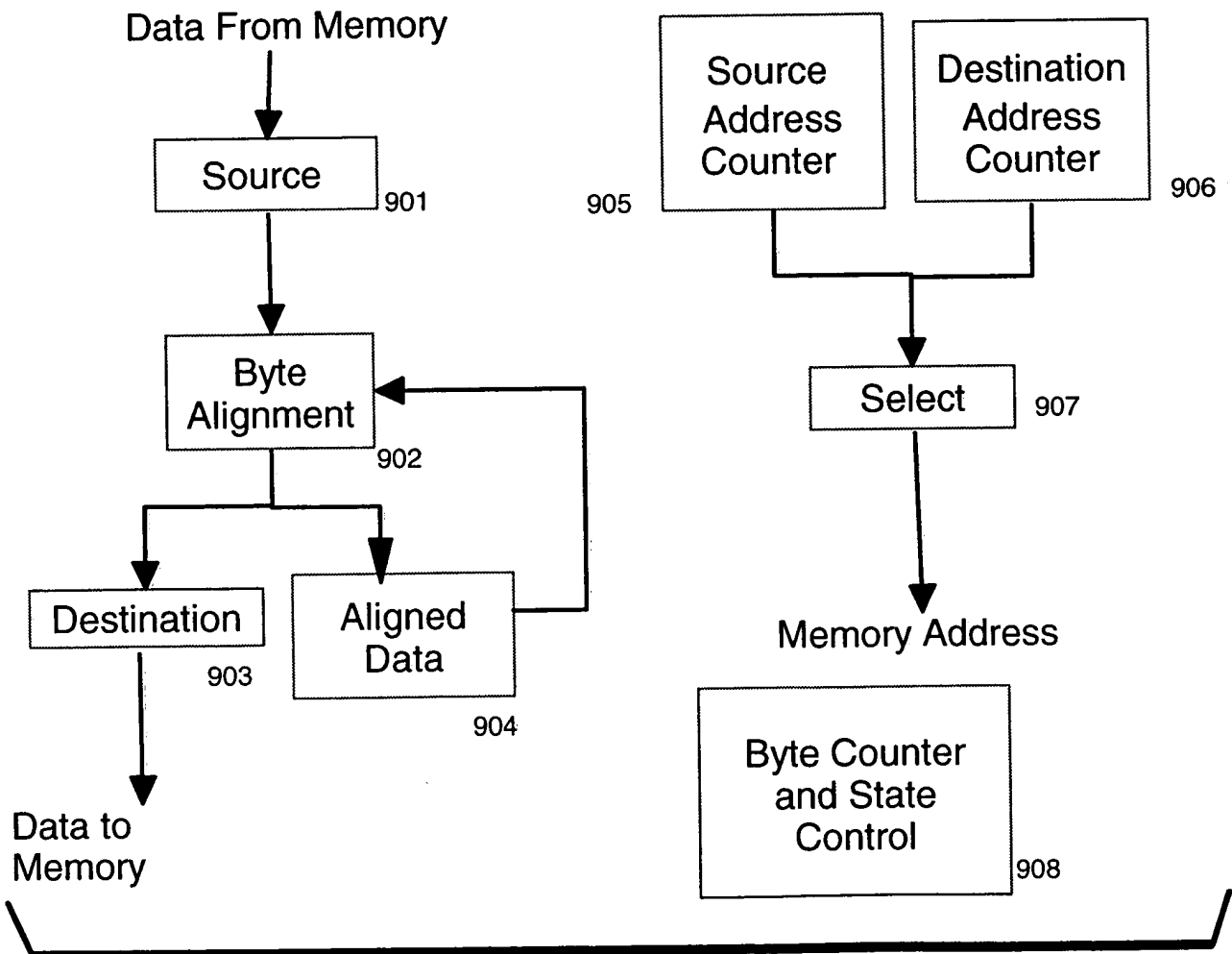
**Fig 8**

Fig 9

1000

MVXL moves the number of bytes specified by the count register from the physical address specified by the source register to the physical address specified by the destination register. The instruction is privileged.

(MVCL performs the same function between virtual addresses.) Here the Device Driver loads the register with physical rather than virtual addresses allowing cross partition data movement.

Fig 11

1101. User calls Device Driver
 - Supplies
 - Source Network ID
 - Source Offset
 - Destination Network ID
1102. Device driver transfers addresses to Adapter
1103. Adapter Translates Addresses
 - Looks up Physical Base addresses from ID's (Table Lookup)
 - Obtains Lock and current Destination Offset
 - Adds offsets
 - Checks bounds
1104. Adapter loads count and addresses in registers
1105. Adapter executes Data Move
1106. Adapter Frees Lock
1107. Adapter notifies device Driver which "Returns" to user

Fig 12

1201. User calls Device Driver
 - Supplies
 - Source Network ID
 - Source Offset
 - Destination Network ID
1202. Device driver sends addresses to adapter
1203. Adapter Translates
 - Looks up Physical Base addresses from ID's (Table Lookup)
 - Obtains Lock and current Destination Offset*
 - Adds offsets
 - Checks bounds
 - Returns Lock and Physical addresses to Device Driver
1204. Device Driver executes Data Move
1205. Device Driver Frees Lock
1206. Device Driver Returns

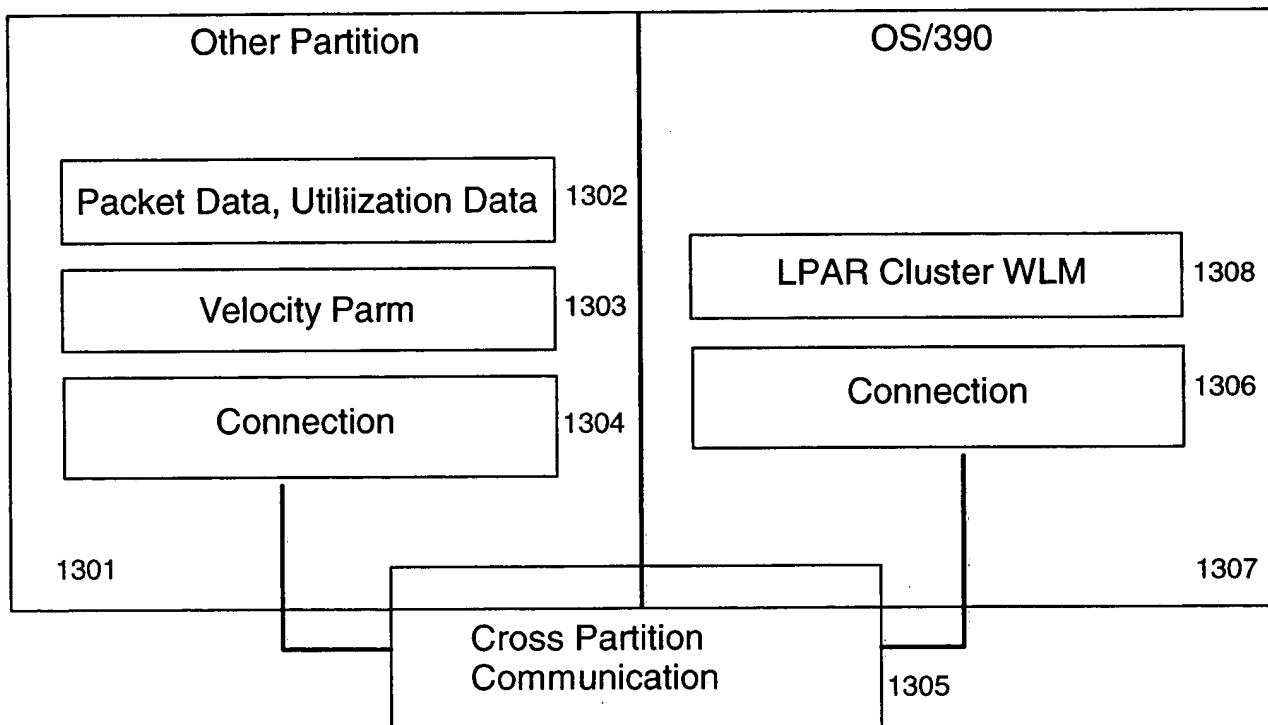
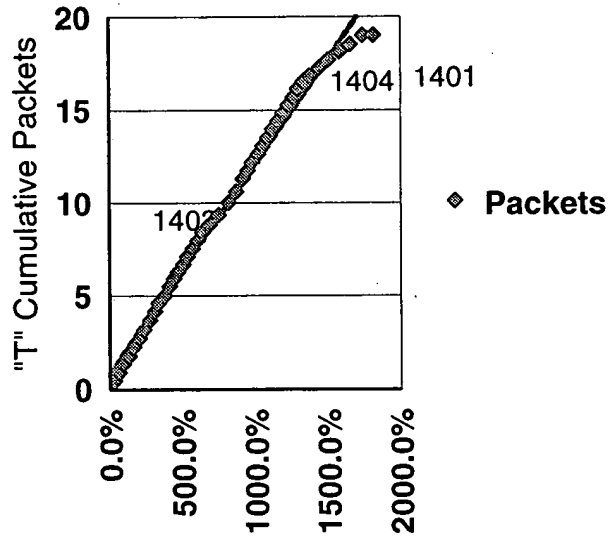


Fig 14

Cum Packets v CPU



"C"- Cumulative CPU

R-square = 0.989 # pts = 43
 $y = 0.802 + 1.13x$

$S = dT/dC$

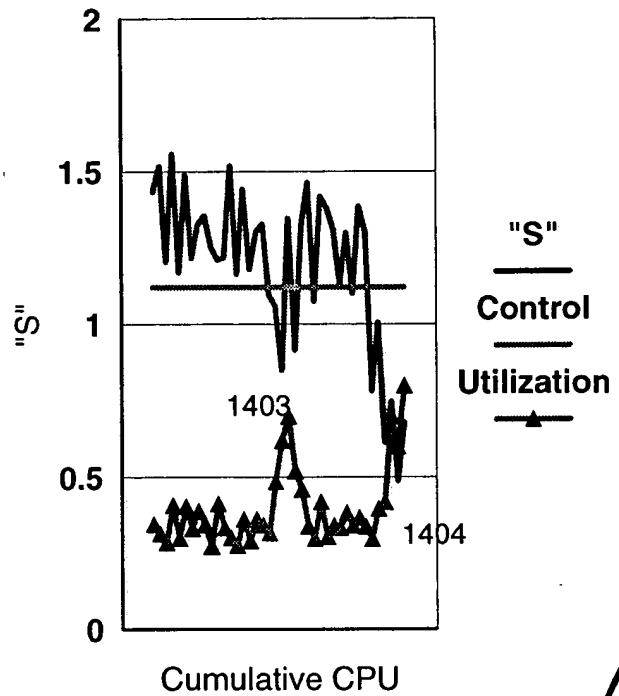


Fig 15

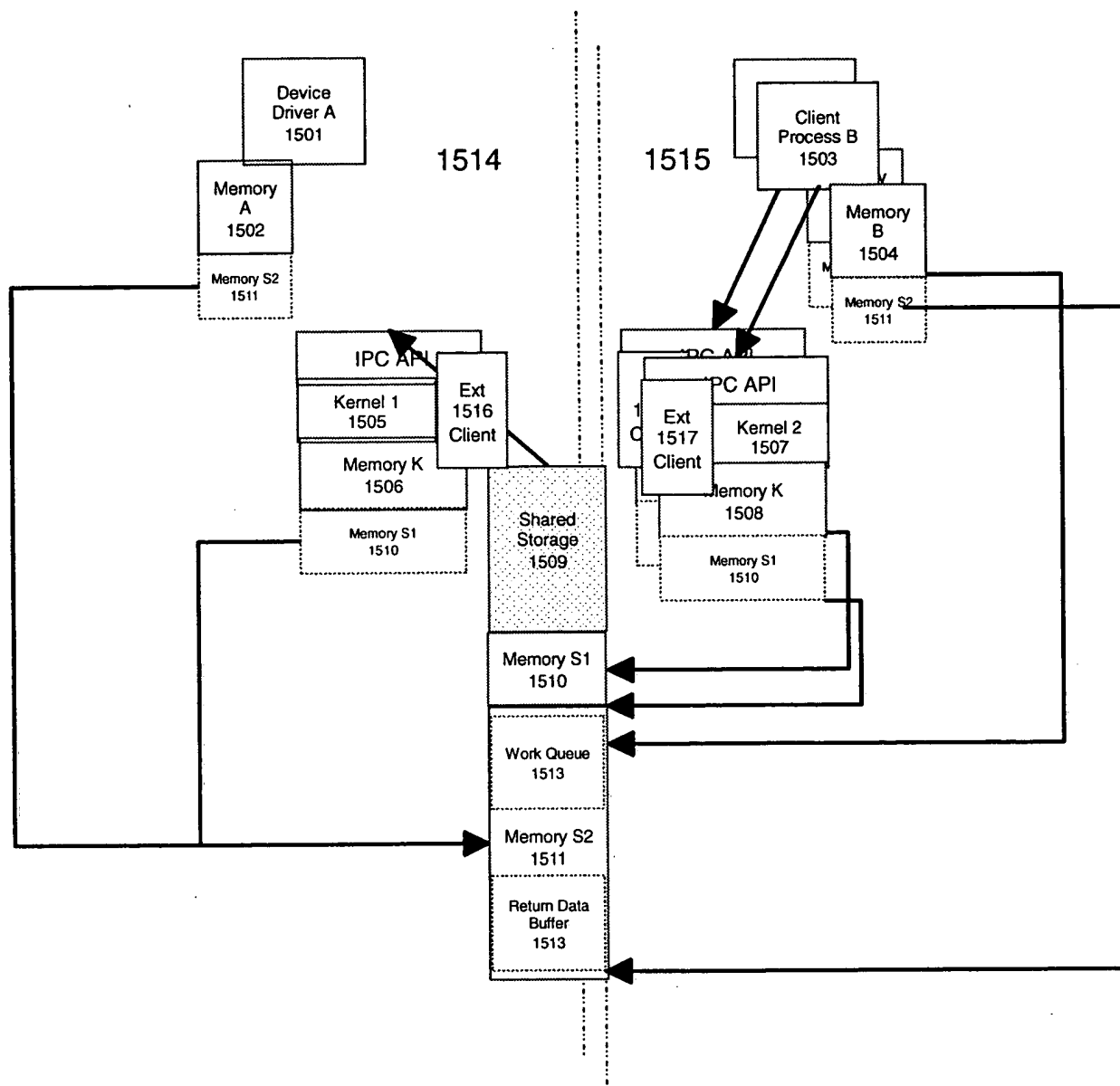


Fig 16